

METHOD AND SYSTEM FOR LONG-TERM UPDATE AND EDIT CONTROL IN A DATABASE SYSTEM

FIELD OF THE INVENTION

The present invention relates to database systems, and more particularly to ensuring data integrity through long term, update and edit control in a database system.

BACKGROUND OF THE INVENTION

Just as computers have become more and more prevalent in everyday life, networks of linked computers have become important in distributing information amongst computer users. Many computer systems are organized according to a client/server metaphor. Generally, in client/server computing, end users are each provided with a desktop computer or terminal known as a "client." The clients are connected using a network to another computer known as a "server", because its general function is to serve or fulfill requests submitted by clients. Application programs running on the clients prepare requests and transmit them to the server over the network. A `network` of computers can be any number of computers that are able to exchange information with one another. The computers may be arranged in any configuration and may be located in the same room or in different countries, so long as there is some way to connect them together (for example, by telephone lines or other communication systems) so they can exchange information. Just as computers may be connected together to make up a network, networks may also be connected together through tools known as bridges and gateways. These tools allow a computer in one network to exchange information with a computer in another network.

Of particular interest in today's computing environment are relational database applications. Relational DataBase Management System (RDBMS) software using a Structured Query Language (SQL) interface is well known in the art. The SQL interface has evolved into a standard language for RDBMS software and has been adopted as such by both the American National Standards Organization (ANSI) and the International Standards Organization (ISO).

In RDBMS software, all data is externally structured into tables. The SQL interface allows users to formulate relational operations on the tables either interactively, in batch files, or embedded in host languages such as C, COBOL, etc. Operators are provided in SQL that allow the user to manipulate the data, wherein each operator operates on either one or two tables and produces a new table as a result. The power of SQL lies in its ability to link information from multiple tables or views together to perform complex sets of procedures with a single statement.

The power of being able to gather, store, and relate information in database systems and then operate on that information through SQL allows for an almost limitless range of applications for such technology. Additionally, the ability to include opaque data types (e.g., character large objects (CLOBs)) along with basic data types (e.g., integers, strings, and decimals) has increased further the capabilities of database systems. With the advent of CLOBs, LOBs, and other opaque types, such as allowed through DB2IMAGE, DB2AUDIO and DB2VIDEO, a problem with ensuring data integrity of these data types has arisen. Due to their nature as more complex data, generally, access for update and edit take longer for opaque data types than for basic data types. Unfortunately, during the long term, extended periods of access, multiple users performing separate edits may diminish the ability to ensure data

integrity of these data types in the database. Accordingly, a need exists for long term, update and edit control of opaque data types in a database system. The present invention addresses such a need.

SUMMARY OF THE INVENTION

Aspects for allowing long term, update and edit control in a database system are described. The aspects include providing library control functions in a database system, and utilizing the library control functions via structured query language statements to ensure data integrity of opaque data types in the database system. Provision of the library control functions further includes providing a checkout function and creating a set of update and delete triggers in correspondence with the checkout function.

Through the present invention, data integrity of opaque data types in a database system is more readily assured. The present invention advantageously employs functions that allow update and edit control to be realized for opaque data types in a straightforward and efficient manner. These functions further allow serialization of work on complex data types, including workflow document data types. These and other advantages of the aspects of the present invention will be more fully understood in conjunction with the following detailed description and accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 illustrates an overall block diagram of a computer system network in accordance with the present invention.

Figure 2 illustrates a diagram representation of a database system environment in accordance with the present invention.

Figure 3 illustrates a block flow diagram for long term, update and edit control in a database system in accordance with the present invention.

Figure 4 illustrates a block flow diagram for implementing the control features of the present invention in a workflow document embodiment.

DETAILED DESCRIPTION

The present invention relates to ensuring data integrity through long term, update and edit control in a database system. The following description is presented to enable one of ordinary skill in the art to make and use the invention and is provided in the context of a patent application and its requirements. Various modifications to the preferred embodiment and the generic principles and features described herein will be readily apparent to those skilled in the art. Thus, the present invention is not intended to be limited to the embodiment shown but is to be accorded the widest scope consistent with the principles and features described herein.

As shown in Figure 1, a plurality of computer systems 1a, 1b, 1c is interconnected via a network 2 (which could be the public Internet or a private intra-corporate Intranet or wide area network). It should be appreciated that although FIG. 1 illustrates a network of computer systems, this is meant as exemplary and not restrictive of the type of environment suitable for the aspects of the present invention. Thus, the aspects may also be provided within a single computing system environment. Accordingly, one (1c) of the computer systems is shown expanded for further illustration.

Computer system 1c has a processor 13 for controlling the overall operation of the computer system 1c, a high speed cache memory 12, a long-term storage device 14 (e.g., hard disk drive), and a database management system 15, e.g., an RDBMS system, such as DB2.

5 In accordance with the present invention, functions are provided that allow long term, update and edit control of opaque data types. These functions are integrated into the database system 15 to allow access via SQL statements executed in the database system 15. Preferably, these functions are provided as user-defined functions. A user-defined function (UDF) generally refers to a function that is defined to the database management system and can be referenced thereafter in SQL queries. Alternatively, the function 20 may be defined through standard techniques as a built-in function within a database system.

Referring to the diagrams of Figure 2 and Figure 3, in accordance with the present invention, a database system 15 is enabled for library control functions 20 for existing and new data in the database system 15 (step 30), which creates a set of update and delete triggers. Once the database is enabled for library control functions, table/column pairs are enabled for control under the library control functions (step 32). The library control functions also create a set of side tables for each selected table that contains library-controlled columns. The side tables are used for controlling update and delete access to the columns. Thus, if the selected table has an identity column, then the library control functions use the identity and column intersection as a key to identify the data cell. If the selected table does not have an identity column, then the library functions alter the user table and create an identity column. Following enablement, utilization of the library control functions occurs (step 34). Thus, a user invokes the library control functions via a select statement from a client application 26

which will checkout (Lock) or checkin (Unlock) a cell or set of cells in a managed database table.

By way of example, suppose an advertising firm had a table that contained images as Logos and Documents describing these logo images, where the column for the Logos and the column for the Documents were under control of the library control functions. A SQL statement for checking out the logo images and the associated document would be suitably represented by:

```
SELECT checkout(logo), checkout(document) FROM hotmedia WHERE company =  
'IBM'
```

In this example, the statement would create a set of entries in the control side tables for the hotmedia table for each row-column pair where the company was equal to 'IBM'. The entries in the control side table would contain the SQL identifier (SQLID) of the user as the identity, and the column name. The column name and identity would be the primary key, which would prevent the cell from being checked out to another user.

If the checkout is successful, then the user would be free to update the checked-out cell with new or modified data. With the control side table, a row can only be deleted if no cells are checked-out, or the only checked-out cells are checked-out to the user performing the deletion. The action of deleting the row also removes any control entries from the control side table.

Alternatively, when the user with the lock updates the item in the row, the control entry is removed from the control side table, making the item available for another user to check out (i.e., it is automatically checked-in on update). Having an item checked-out does not prevent

anyone from querying the item. It only prevents changes (i.e., deletes and updates). Thus, control is achieved over updates and edits for opaque data types.

The control over update and edit of opaque data types in accordance with the present invention allows for serialization of work on complex data objects. In this manner, workflow documents are included as a type of opaque data type. Current workflow systems use dedicated connect clients that need to be connected to the workflow server in order to accomplish their tasks. The dedicated connection restricts the ability to dynamically expand the workflow system in a work environment, such as across the Internet. In order to provide a more expandable workflow system, the aspects of library control functions are combined with an XML Extender feature of DB2, as described with reference to the flow diagram of Figure 4.

The process starts with an XML (extensible markup language) workflow document arriving in an electronic mail queue of a system participating as a workflow system (step 40). The workflow document is decomposed by the XML Extender into the IncomingQueue table of the database (step 42). Triggers on the IncomingQueue table use the workflow type, key, and user-id columns of the table to determine which control table contains the control record that is associated with the request (step 44). This information is used by the triggers to update the control record with the current workflow step and update a workflow data table with the incoming data (step 46). The trigger also uses the state information in the control record to do a lookup in the workflow table to determine the next step in the workflow and to identify a handler for that next step (step 48). Once the triggers have updated the control record, an after trigger fires a stored procedure, which checks out the workflow control record, composes an XML document using the XML Extender, and sends that to the appropriate handler for processing the next step (step 50).

Through the present invention, data integrity of opaque data types in a database system is more readily assured. The present invention advantageously employs functions that allow update and edit control to be realized for opaque data types in a straightforward and efficient manner. These functions further allow serialization of work on complex data types, including workflow document data types.

Although the present invention has been described in accordance with the embodiments shown, one of ordinary skill in the art will readily recognize that there could be variations to the embodiments and those variations would be within the spirit and scope of the present invention. Accordingly, many modifications may be made by one of ordinary skill in the art without departing from the spirit and scope of the appended claims.